USING RGB COLOUR COMBINATION IN COLOURED QUICK RESPONSE (QR) CODE ALGORITHM TO ENHANCE QR CODE CAPACITY

AZIZI BIN ABAS

DOCTOR OF PHILOSOPHY
UNIVERSITI UTARA MALAYSIA
2018

USING RGB COLOUR COMBINATION IN COLOURED QUICK RESPONSE (QR) CODE ALGORITHM TO ENHANCE QR CODE CAPACITY

AZIZI BIN ABAS

Thesis Submitted to
Awang Had Salleh Graduate School of Arts and Sciences
Universiti Utara Malaysia,
In Fulfillment of the Requirement for the Degree of Doctor Philosophy

Dissertation

Awang Had Salleh
Graduate School
of Arts And Sciences

Universiti Utara Malaysia

PERAKUAN KERJA TESIS / DISERTASI
(Certification of thesis / dissertation)

Kami, yang bertandatangan, memperakukan bahawa
(We, the undersigned, certify that)

AZIZI ABAS

calon untuk Ijazah **PhD**
(candidate for the degree of)

telah mengemukakan tesis / disertasi yang bertajuk:
(has presented his/her thesis / dissertation of the following title):

"USING RGB COLOUR COMBINATION IN COLOURED QUICK RESPONSE (QR) CODE ALGORITHM TO ENHANCE QR CODE CAPACITY"

seperti yang tercatat di muka surat tajuk dan kulit tesis / disertasi.
(as it appears on the title page and front cover of the thesis / dissertation).

Bahawa tesis/disertasi tersebut boleh diterima dari segi bentuk serta kandungan dan meliputi bidang ilmu dengan memuaskan, sebagaimana yang ditunjukkan oleh calon dalam ujian lisan yang diadakan pada : ***31 Mei 2018.***
That the said thesis/dissertation is acceptable in form and content and displays a satisfactory knowledge of the field of study as demonstrated by the candidate through an oral examination held on: ***May 31, 2018.***

Pengerusi Viva: *(Chairman for VIVA)*	**Assoc. Prof. Dr. Osman Ghazali**	Tandatangan *(Signature)*
Pemeriksa Luar: *(External Examiner)*	**Prof. Dr.Rusli Abdullah**	Tandatangan *(Signature)*
Pemeriksa Luar: *(External Examiner)*	**Assoc. Prof. Dr. Mohd Pouzi Hamzah**	Tandatangan *(Signature)*
Nama Penyelia/Penyelia-penyelia: *(Name of Supervisor/Supervisors)*	**Assoc. Prof. Dr. Yuhanis Yusof**	Tandatangan *(Signature)*
Nama Penyelia/Penyelia-penyelia: *(Name of Supervisor/Supervisors)*	**Dr. Farzana Kabir Ahmad**	Tandatangan *(Signature)*

Tarikh:
(Date) ***May 31, 2018***

Permission to Use

In presenting this thesis in fulfilment of the requirements for a postgraduate degree from Universiti Utara Malaysia, I agree that the Universiti Library may make it freely available for inspection. I further agree that permission for the copying of this thesis in any manner, in whole or in part, for scholarly purpose may be granted by my supervisor(s) or, in their absence, by the Dean of Awang Had Salleh Graduate School of Arts and Sciences. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to Universiti Utara Malaysia for any scholarly use that may be made of any material from my thesis.

Requests for permission to copy or to make other use of materials in this thesis, in whole or in part, should be addressed to:

Dean of Awang Had Salleh Graduate School of Arts and Sciences
UUM College of Arts and Sciences
Universiti Utara Malaysia
06010 UUM Sintok

Abstrak

Kod Respons Pantas (QR) ialah kod bar dua dimensi yang menyimpan aksara dan boleh dibaca oleh mana-mana kamera telefon pintar. Kod QR mempunyai keupayaan untuk mengekod pelbagai format data dan bahasa. Walau bagaimanapun, Kod QR hitam dan putih yang sedia ada menyediakan penyimpanan data yang terhad. Walaupun terdapat penyelidikan mengenai Kod QR berwarna untuk meningkatkan kapasiti penyimpanan, keperluan untuk kapasiti data yang lebih besar oleh pengguna terus meningkat. Oleh itu, tesis ini mencadangkan algoritma Kod QR berwarna yang menggunakan kombinasi warna merah, hijau dan biru (RGB) untuk membolehkan storan data yang lebih besar. Algoritma yang dicadangkan mengintegrasikan penggunaan teknik mampatan, pemultipleksan, dan pelbagai lapis dalam pengekodan dan penyahkodan Kod QR. Tambahan pula, ia juga memperkenalkan algoritma pengekodan/penyahkodan separa yang membolehkan pemanipulasi data. Algoritma yang merangkumi proses pengekodan dan penyahkodan adalah berdasarkan teknik warna RGB, yang digunakan untuk membuat Kod QR berwarna berkapasiti tinggi. Ini direalisasikan dalam eksperimen yang menyimpan aksara Kod Piawai Amerika bagi Saling Tukar Maklumat (ASCII). Aksara teks ASCII digunakan sebagai input dan prestasi diukur dengan bilangan aksara yang boleh disimpan di dalam Kod QR hitam dan putih versi 40 (iaitu tanda aras) dan juga Kod QR berwarna. Metrik eksperimen lain termasuk peratusan aksara yang hilang, bilangan Kod QR yang dihasilkan, dan masa berlalu untuk membuat Kod QR. Hasil simulasi menunjukkan bahawa algoritma yang dicadangkan menyimpan 29 kali lebih banyak aksara daripada Kod QR hitam dan putih dan 9 kali lebih banyak daripada Kod QR berwarna lain. Oleh itu, ini menunjukkan bahawa Kod QR yang berwarna mempunyai potensi untuk menjadi penyimpanan mini data kerana ia tidak bergantung kepada sambungan internet.

Kata kunci: Kod respons pantas, Kod bar, Pencapaian maklumat, Penyimpanan data, Warna RGB

Abstract

A Quick Response (QR) Code is a two-dimensional barcode that stores characters and can be read by any smartphone camera. The QR code has the capability to encode various data formats and languages; nevertheless, existing black and white QR code offers limited data storage. Even though there exist research on coloured QR Code to increase the storage capacity, requirement for larger data capacity by end user keep increasing. Hence, this thesis proposes a coloured QR Code algorithm which utilizes RGB colour combination to allow a larger data storage. The proposed algorithm integrates the use of compression, multiplexing, and multilayer techniques in encoding and decoding the QR code. Furthermore, it also introduces a partial encoding/decoding algorithm that allows the stored data to be manipulated. The algorithm that includes encoding and decoding processes is based on the red, green, and blue (RGB) colour techniques, which are used to create high capacity coloured QR code. This is realised in the experiments that store American Standard Code for Information Interchange (ASCII) characters. The ASCII text characters are used as an input and performance is measured by the number of characters that can be stored in a single black and white QR code version 40 (i.e. the benchmark) and also the coloured QR code. Other experiment metrics include percentage of missing characters, number of produced QR code, and elapsed time to create the QR code. Simulation results indicate that the proposed algorithm stores 29 times more characters than the black and white QR code and 9 times more than other coloured QR code. Hence, this shows that the coloured QR Code has the potential of becoming a useful mini-data storage as it does not rely on internet connection.

Keywords: Quick Response Code, Barcode, Information retrieval, Data storage; RGB colours.

Acknowledgement

First of all, I would like to thank my supervisor, Assoc. Prof. Dr. Yuhanis and Dr. Fauzana Kabir Ahmad for giving me the opportunity to pursue this long and rewarding journey, and for their help and guidance.

Most of all, I would like to thank my mother, Puteh Ismail, for her unconditional support in every way and for her trust and love. To my father, Abas Ismail, in which I undoubtedly see myself every day and who from heaven helped me to achieve what I once saw so far away.

To my wife Zuraida Saad, my daughter Alia Qistina and my son Adib Qayyum, thank you for your understanding, for loving me, and for being there for me all these years. This was a very special period in my life in which I have great successes and catastrophic failures, in which I learned about myself and the others, and in which I was reminded that what matters is always the journey and not the destination. None of these could have been possible without all of you, and for that I just would like to say thank you. Also, to all my friends in UUM who never let me forget.

Let us close this chapter today and start a new one in this story, without forgetting what I learned, what I am, and what I want to become.

Table of Contents

List of Tables

List of Figures

List of Appendices

List of Abbreviations

1D	One-dimensional
2D	Two-dimensional
3D	Three-dimensional
ANSI	American National Standard Institute
ASCII	American Standard Code for Information Interchange
CIAL	Content Idea Asia Limited
CMY	Cyan, Magenta, and Yellow
CMYK	Cyan Magenta Yellow and Key (Black)
CQR	Colour Quick Response
CQRC	Colour Quick Response Code
GZip	GNU Zip (Not Unix Zip)
HCC2D	High Capacity Coloured Two Dimensional
HCCB	High Capacity Colour Barcode
ISO	International Organization for Standardization
LED	Light Emitting Diode

LZW	Lempel–Ziv–Welch
MATLAB	Matrix Laboratory
PM	Paper Memory
RGB	Red Green Blue
RO	Research Objective
RQ	Research Question
URL	Uniform Resource Locator
UTF	Unicode Transformation Format
QR code	Quick Response Code
ZIP	Compressed File Archive

CHAPTER ONE

INTRODUCTION

This research is on quick response code technology, which is one of the mechanisms to store information using two dimensional (2D) barcode images. Instead of using only white and black colour modules, this research proposes a coloured code that enables a larger data storage capacity.

1.1 Introduction

Currently, the use of digital media and communications technologies is growing rapidly from time to time. But in the same time, printed documents continue to form a convenient interface for people. A large number of important documents such as identity card, driving licence, passports, and other transaction data are still in printed form. Without exception, some of the printed items are used to tell information about the object or owner. Now in the digital era, one technique or mechanism is needed to interface with the information in the printed items or documents, which can be embedded inside printed objects. Thus, it can save more space in the printed document and it is secure. The data can subsequently be retrieved via a scanner or digital camera that can be aimed at the printed object (Bulan & Sharma, 2011b). In addition, it facilitates users to store data without using an electronic data storage device and saves the area of printed items or documents. The technique or mechanism used to embed digital information inside the printed object must be provided with additional operational features in the applications such as document authentication, meta-data embedding, and document tracking in workflows (Bulan & Sharma, 2011b). The information and methods as mentioned above refer to the use of barcode.

The contents of the thesis is for internal user only

REFERENCES

Abas, A., Yusof, Y., & Ahmad, F. K. (2017). Expanding the data capacity of QR codes using multiple compression algorithms and base64 encode/decode. *Journal of Telecommunication, Electronic and Computer Engineering*, *9*(2–2).

Ahlawat, S., & Rana, C. (2017). A Review on QR Codes : Colored and Image Embedded. *International Journal of Advanced Research in Computer Science*, *8*(5), 410–413.

Anonymous. (2013). PM-Code's world. Retrieved July 7, 2018, from http://pmcode.co-site.jp/

Asare, I. T., & Asare, D. (2015). The Effective Use of Quick Response (QR) Code as a Marketing Tool. *International Journal of Education and Social Science*, *2*(12), 67–73.

Bagherinia, H., & Manduchi, R. (2012). High Information Rate and Efficient Color Barcode Decoding. In *International Workshop on Color and Photometry in Computer Vision (CPVC)* (pp. 1–10).

Bhardwaj, N., Kumar, R., Verma, R., Jindal, A., & Bhondekar, A. P. (2016). Decoding algorithm for color QR code: A mobile scanner application. In *2016 International Conference on Recent Trends in Information Technology, ICRTIT 2016*. https://doi.org/10.1109/ICRTIT.2016.7569561

Bishop, T. (2007). Software notebook: Color is key to Microsoft's next-generation bar code. Retrieved July 7, 2018, from https://www.microsoft.com/en-us/research/project/high-capacity-color-barcodes-hccb/

Blasinski, H., Bulan, O., & Sharma, G. (2013). Per-Colorant-Channel Color Barcodes for Mobile Applications : An Interference Cancellation. In *IEEE Transaction on Image processing* (Vol. 22, pp. 1498–1511).

Boob, A., Shinde, A., Rathod, D., & Gaikwad, A. (2014). Qr Code Based Mobile App and Business Process Integration. *International Journal of Multidisciplinary and Current Research*, *2*(Sept and Oct 2014), 1014–1017.

Brititsh Standards. (2009). *Information technology : automatic identification and data capture techniques, QR code 2005 bar code symbology specification. ISO Standards*.

Bulan, O., & Sharma, G. (2011a). High Capacity Color Barcodes : Per Channel Data Encoding via Orientation Modulation in. *IEEE Transactions on Image Processing*, *20*(5), 1337–1350.

Bulan, O., & Sharma, G. (2011b). *High Capacity Data Embedding For Printed Documents*. University of Rochester.

Bunma, D., & Vongpradhip, S. (2014). Using augment reality to increase capacity in QR code. In *4th International Conference on Digital Information and Communication Technology and Its Applications, DICTAP 2014* (pp. 440–443). https://doi.org/10.1109/DICTAP.2014.6821727

Chandran, A. (2014). Review on Color Qr Codes : Decoding Challenges. *International Journal of Engineering Research & Technology (IJERT)*, *3*(4), 848–851.

Chang, J. H. (2014). An introduction to using QR codes in scholarly journals. *Science Editing*, *1*(2), 113–117. https://doi.org/10.6087/kcse.2014.1.113

Charoensiriwath, C., Surasvadi, N., Pongnumkul, S., & Pholprasit, T. (2015). Applying QR code and mobile application to improve service process in Thai hospital. In *Proceedings of the 2015 12th International Joint Conference on Computer Science and Software Engineering, JCSSE 2015* (pp. 114–119). https://doi.org/10.1109/JCSSE.2015.7219781

Chiang, J. S., Li, H. T., Hsia, C. H., Wu, P. H., & Hsieh, C. F. (2013). High density QR code with multi-view scheme. In *Proceedings of the International Symposium on Consumer Electronics, ISCE* (Vol. 49, pp. 49–50). https://doi.org/10.1109/ISCE.2013.6570246

Chuang, J.-C., Hu, Y.-C., & Ko, H.-J. (2010). A Novel Secret Sharing Technique Using QR Code. *International Journal of Image Processing (IJIP)*, *4*(5), 468–475.

Coleman, J. (2011). QR Codes: What Are They and Why Should You Care? In *Kansas Library Association College and University Libraries Section Proceedings* (Vol. 1, pp. 16–23). https://doi.org/10.4148/culs.v1i0.1355

Commission, & International Organization for Standardization., I. E. (2000). *Information technology -- automatic identification and data capture techniques -- bar code symbology -- QR code*. Geneva: Geneva : ISO : IEC, 2000.

Convert Words to Pages. (2016). Retrieved May 7, 2017, from http://www.wordstopages.com/

Čović, Z., & Šimon, J. (2016). Usage of QR codes in promotion on social networks. In *Proceedings of 2016 International Conference on Smart Systems and Technologies, SST 2016* (pp. 123–127). https://doi.org/10.1109/SST.2016.7765645

Dagan, I., Binyamin, G., & Eilam, A. (2016). Delivery of QR Codes to Cellular Phones through Data Embedding in Audio. In *ISCEE International Conference on the Science of Electrical Engineering* (pp. 3–6).

Demir, S., Kaynak, R., & Demir, K. A. (2015). Usage Level and Future Intent of Use of Quick Response (QR) Codes for Mobile Marketing among College Students in Turkey. *Procedia - Social and Behavioral Sciences*, *181*, 405–413. https://doi.org/10.1016/j.sbspro.2015.04.903

Denso-Wave. (2015). QR Code Standardization. Retrieved July 7, 2018, from http://www.qrcode.com/en/about/standards.html

Denso, A. (2011). *Qr code essentials*. *Retrieved fr om http://www. nacs. org/LinkClick. aspx*. Retrieved from http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:QR+Code+®+Essentials#0

Denso Wave. (2014). Denso Wave, The Inventor of QR Code. Retrieved July 7, 2018, from https://www.denso-autoid-eu.com/en/about-us/20-years-qr-code.html

Dita, I., Otesteanu, M., & Quint, F. (2011). Data Matrix Code - A Reliable Optical Identification of Microelectronic Components. *2011 IEEE 17th International Symposium for Design and Technology in Electronic Packaging (SIITME)*, *1*(1), 39–44.

Donald D. Hearn, M. Pauline Baker, W. C. (2010). *Computer Graphics with Open GL (4th Edition)* (4th ed.). Pearson.

Donoho, D. L., Vetterli, M., Devore, R. A., & Daubechies, I. (1998). Data Compression and Harmonic Analysis. In *IEEE Transaction on Information Theory* (Vol. 44, pp. 2435–2476).

Falas, T., & Kashani, H. (1994). Two-Dimensional Bar-code Decoding with Camera-Equipped Mobile Phones. In *Pervasive Computing and Communications Workshops, 2007. PerCom Workshops '07. Fifth Annual IEEE International Conference* (pp. 2–5).

Farizshah, M., & Abd Jalil, K. (2012). The Embedding of Arabic Characters in QR Code. In *International Conference Open Systems (ICOS), 2012*.

Feng, X., & Zheng, H. (2010a). Design and realization of 2D color barcode with high compression ratio. In *International Conference on Computer Design and Applications, ICCDA 2010* (Vol. 1, pp. 314–317). https://doi.org/10.1109/ICCDA.2010.5540872

Feng, X., & Zheng, H. (2010b). Design and realization of 2D color barcode with high compression ratio. In *International Conference on Computer Design and Applications, ICCDA 2010* (Vol. 1, p. 4). https://doi.org/10.1109/ICCDA.2010.5540872

Ferreira, R. a. S., & André, P. S. (2014). Colour multiplexing of quick-response (QR) codes. *Electronics Letters*, *50*(24), 1828–1830. https://doi.org/10.1049/el.2014.2501

Fried, I. (2007, July 16). Microsoft gives bar codes a splash of color. *ZDNet*. Retrieved from https://www.zdnet.com/article/microsoft-gives-bar-codes-a-splash-of-color/

Frost, C., Mammarella, M., Kohler, E., Reyes, A. D. L., Hovsepian, S., Matsuoka, A., & Zhang, L. (2007). Generalized File System Dependencies. *SOSP'07*, *1*, 14.

Fukuchi, K. (2010). Libqrencode, a c library for encoding data in a qr code symbol. Retrieved July 7, 2018, from https://fukuchi.org/works/qrencode/

Galiyawala, H. J., & Pandya, K. H. (2014). To Increase Data Capacity of QR Code Using Multiplexing with Color Coding : An example of Embedding Speech Signal in QR Code. In *2014 Annual IEEE India Conference (INDICON)* (pp. 2–7).

Galiyawala, H. J., & Pandya, K. H. (2015). To increase data capacity of QR code using multiplexing with color coding: An example of embedding speech signal in QR code. In *11th IEEE India Conference: Emerging Trends and Innovation in Technology, INDICON 2014* (pp. 2–7). https://doi.org/10.1109/INDICON.2014.7030441

Garateguy, G. J. (2014). *Optimal Embedding of QR codes into Color, Gray Scale and Binary Images*. University of Delaware. Retrieved from http://udspace.udel.edu/handle/19716/13350

Garateguy, G. J., Member, S., Arce, G. R., Lau, D. L., Member, S., & Villarreal, O. P. (2014). QR Images : Optimized Image Embedding in QR Codes. *IEEE Transactions on Image Processing*, *23*(7), 2842–2853.

Gerstner, T., Decarlo, D., Alexa, M., Finkelstein, A., Gingold, Y., & Nealen, A. (2013). Pixelated image abstraction with integrated user constraints. *Computers and Graphics (Pergamon)*, *37*(5), 333–347. https://doi.org/10.1016/j.cag.2012.12.007

Goel, S., & Singh, A. K. (2014). Cost Minimization by QR Code Compression. *International Journal of Computer Trends and Technology (IJCTT)*, *15*(4), 157–161.

Grillo, A., Lentini, A., Querini, M., & Italiano, G. F. (2010). High capacity colored two dimensional codes. *Computer Science and Information Technology (IMCSIT), Proceedings of the 2010 International Multiconference on*, *5*(1), 709–716. https://doi.org/10.1109/IMCSIT.2010.5679869

Gunawi, H. S., Prabhakaran, V., Krishnan, S., Arpaci-Dusseau, A. C., & Arpaci-Dusseau, R. H. (2007). Improving File System Reliability with I / O Shepherding. In *21st ACM Symposium on Operating Systems Principles* (p. 14).

Gutierrez, F., Abud, M. A., Vera, F., & Sanchez, J. A. (2013). Application of contextual QR codes to augmented reality technologies. In *23rd International Conference on Electronics, Communications and Computing, CONIELECOMP 2013* (pp. 264–269). https://doi.org/10.1109/CONIELECOMP.2013.6525798

Guwalani, P., Kala, M., Chandrashekar, R., Shinde, J., & Mane, D. (2014). Image File Security using Base-64 Algorithm. *International Journal Computer Technology & Applications*, *5*(6), 1892–1895.

Hahn, H. I., & Joung, J. K. (2002). Implementation of Algorithm to Decode Two-Dimensional Barcode PDF-417. In *ICSP'O2 Proceedings*.

Hajduk, V., Broda, M., Kováþ, O., & Levický, D. (2016). Image steganography with using QR code and cryptography. In *26th Conference Radioelektronika 2016* (pp. 350–353). https://doi.org/10.1109/RADIOELEK.2016.7477370

Harish, N., & Gurav, S. (2014). Embedding a Large Information In QR Code Using Multiplexing Technique. *Taraksh Journal of Communications*, *1*(6), 6–9.

Hashem, I. A. T., Yaqoob, I., Anuar, N. B., Mokhtar, S., Gani, A., & Ullah Khan, S. (2015). The rise of "big data" on cloud computing: Review and open research issues. *Information Systems*, *47*, 98–115. https://doi.org/10.1016/j.is.2014.07.006

He, D., Sun, Y., Jia, Z., Yu, X., Guo, W., He, W., … Lu, X. (2010). A Proposal of Substitute for Base85 / 64 – Base91. In *The SUMMER 8th International Conference on Computing, Communications and Control Technologies 2010* (Vol. 85, pp. 4–7).

Husain, A., Bakhtiari, M., & Zainal, A. (2014). Printed document integrity verification using barcode. *Jurnal Teknologi (Sciences and Engineering)*, *70*(1), 99–106. https://doi.org/10.11113/jt.v70.2857

Intermec Technologies Corporation. (2007, November). The 2D Revolution How evolving business needs and improved technology are driving explosive growth in two-dimensional bar coding, 2. Retrieved from https://www.technologynetworks.com/tn/go/lc/view-white-paper-230951

Jahagirdar, K. S., & Borse, S. B. (2015). QR Code with Colored Image. *International Journal of Computer Applications*, *115*(16), 38–41.

Jancke, G. (2015). High Capacity Color Barcodes (HCCB). Retrieved July 28, 2018, from https://www.microsoft.com/en-us/research/project/high-capacity-color-barcodes-hccb/

Jennifer Farley. (2010). A Short Guide To Color Models. *SitePoint*, 1. Retrieved from https://www.sitepoint.com/a-short-guide-to-color-models/

Kajaree, D., & Behera, R. . (2017). A Survey on Machine Learning: Concept, Algorithms and Applications. *International Journal of Innovative Research in Computer and Communication Engineering*, *5*(2), 1302–1309. https://doi.org/10.15680/IJIRCCE.2017.

Kan, T., Teng, C.-H., & Chou, W.-S. (2009). Applying QR code in augmented reality applications. In *Proceedings of the 8th International Conference on Virtual Reality Continuum and its Applications in Industry - VRCAI '09* (Vol. 1, p. 253). https://doi.org/10.1145/1670252.1670305

Kato, H., & Tan, K. T. (2005). 2D barcodes for mobile phones. In *Proceeding Mobile Technology, Applications and Systems, 2005 2nd International Conference* (p. 8).

Kattan, A., & Poli, R. (2008). Evolutionary Lossless Compression with GP-ZIP. In *IEEE Congress on Evolutionary Computation, CEC 2008* (pp. 335–364).

Kieseberg, P., Leithner, M., Mulazzani, M., Munroe, L., Schrittwieser, S., Sinha, M., & Weippl, E. (2010). QR code security. In *Proceedings of the 8th International Conference on Advances in Mobile Computing and Multimedia - MoMM '10* (p. 430). https://doi.org/10.1145/1971519.1971593

Kim, H. M., Kim, W., & Cho, D. (2004). A New Color Transforming for RGB Coding. In *ICIP '04. 2004 International Conference* (pp. 107–110).

Kingsley G. Morse Jr. (2005). Compression Tools Compared. Retrieved May 2, 2017, from http://www.linuxjournal.com/article/8051?page=0,0

Kumar, K., Sharma, P., & Singh, A. K. (2012). Configuring the System to Share Internet from Single User to Multi-user with Single Internet Dongle. *International Journal of Soft Computing and Engineering (IJSCE)*, (4), 32–35.

Kumaraguru, S., & Prof Dr D. S. Bormane. (2012). Identification of QR Code based on Pattern Recognition with Mobile Phones. *International Journal of Modern Engineering Research (IJMER)*, *2*(5), 3544–3547.

Li, M., Cao, P., Yu, L., Liuping, F., Chen, J., & Jing, W. (2017). The research of information hiding and extraction based on QR code positioning function. In *2nd IEEE International Conference on Computer and Communications, ICCC 2016 - Proceedings* (pp. 589–593). https://doi.org/10.1109/CompComm.2016.7924769

Li, Z., Chen, Z., Sudarshan, M. S., & Yuanyuan, Z. (2004). C-Miner: Mining Block Correlations in Storage Systems. In *Proceedings of the Third USENIX Conference on File and Storage Technologies* (p. 14).

Liao Zhao-lai, Huang Ting-lei, Wang Rui, Z. X. (2010). A Method of Image Analysis for QR Code Recognition. In *2010 International Conference on Intelligent Computing and Integrated Systems* (pp. 250–253). https://doi.org/10.1109/ICISS.2010.5657187

Lin, J., & Fuh, C. (2013). 2D Barcode Image Decoding. *The Scientific World Journal*, *2013*(3), 1.

Liu, Y., Yang, J., & Liu, M. (2008). Recognition of QR Code with mobile phones. *2008 Chinese Control and Decision Conference*, 203–206. https://doi.org/10.1109/CCDC.2008.4597299

Liu, Z., Zheng, H., & Jia, H. (2009). Design and implementation of color two-dimension barcode with high compression ratio for Chinese characters. In *Proceedings - 2009 International Conference on Information Engineering and Computer Science, ICIECS 2009*. https://doi.org/10.1109/ICIECS.2009.5363553

Luo, M., Wang, S., & Lin, P. Y. (2016). QR code steganography mechanism with high capacity. In *2016 International Conference On Communication Problem-Solving (ICCP)* (pp. 1–2). https://doi.org/10.1109/ICCPS.2016.7751131

Lyons, S. (2009). *Two-Dimensional Barcodes for Mobile Phones*. University of Toronto.

Magadum, B. (2017). Data security in QR code using Steganography. *International Journal of Innovative Research in Computer and Communication Engineering*, *5*(5), 10058–10063.

Majumdar, S., Maiti, A., Bhattacharyya, B., & Nath, A. (2015). A new encrypted Data hiding algorithm inside a QR Code ™ implemented for an Android Smartphone system : S _ QR algorithm Introduction : *International Journal of Innovative Research in Advanced Engineering (IJIRAE)*, *2*(4), 40–46.

Marktscheffel, T., Gottschlich, W., Popp, W., Werli, P., Fink, S. D., Bilzhause, A., & Meer, H. de. (2016). QR code based mutual authentication protocol for Internet of Things. In *2016 IEEE 17th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)* (pp. 1–6). https://doi.org/10.1109/WoWMoM.2016.7523562

Melgar, M. E. V., & Santander, L. M. (2016). Channel capacity analysis of 2D barcodes: QR Code and CQR Code-5. In *2016 IEEE Colombian Conference on Communications and Computing, COLCOM 2016 - Conference Proceedings* (Vol. 1). https://doi.org/10.1109/ColComCon.2016.7516376

Meyer, D. T., Aggarwal, G., Cully, B., Lefebvre, G., Feeley, M. J., Hutchinson, N. C., & Warfield, A. (2008). Parallax : Virtual Disks for Virtual Machines. In *EuroSys08* (p. 44).

Nandhini. (2017). Performance Evaluation of Embedded Color QR Code on Logos. In *2017 Third International Conference On Science Technology Engineering and Management (ICONSTEM)* (pp. 1009–1014).

Narayanan, D., Donnelly, A., & Rowstron, A. (2008). Write Off-Loading : Practical Power Management for Enterprise Storage. *ACM Transactions on Storage (TOS)*, *4*(3), 15.

Nikolaos, T., & Kiyoshi, T. (2010). QR-Code Calibration for Mobile Augmented Reality Applications. In *SIGGRAPH 2010* (p. 4503).

Nurwono, K., & Kosala, R. (2009). Color quick response code for mobile content distribution. In *Proceedings of MoMM2009* (pp. 267–271). Retrieved from http://dl.acm.org/citation.cfm?id=1821799

Okazaki, S., Navarro, a, & Campo, S. (2013). Cross-media integration of Qr code: a preliminary exploration. *Journal of Electronic Commerce …*, *14*(2), 137–148. Retrieved from http://csulb.edu/journals/jecr/issues/20132/paper1.pdf

Online Reference and Tool. (2012). Retrieved July 7, 2018, from http://www.rapidtables.com/web/color/RGB_Color.htm

Oswal, S., Singh, A., & Kumari, K. (2016). Deflate compression algorithm. *International Journal of Engineering Research and General Science 2016*, *4*(1), 430–436.

Pandya, K. H., & Galiyawala, H. J. (2014). A Survey on QR Codes : in context of Research and Application. *International Journal of Emerging Technology and Advanced Engineering*, *4*(3), 258–262.

Parikh, D., & Jancke, G. (2008). Localization and Segmentation of A 2D High Capacity Color Barcode Microsoft ' s HCCB Barcode localization. In *Applications of Computer Vision, 2008. WACV 2008. IEEE Workshop on* (Vol. 1, p. 6).

Pascale, D. (2003). *A Review of RGB Color Spaces*. (D. Pascale, Ed.) (1st ed.). Montreal, Canada: The BabelColor Company.

Pathak, S., Singh, S., Singh, S., Jain, M., Sharma, A., & 5. (2011). Data Compression Scheme of Dynamic Huffman Code for Different Languages. *International Conference on Information and Network Technology 2011*, *4*(September 2010), 201–206.

Pillai, P. N., & Naresh, K. (2014). Improving the Capacity of QR Code by Using Color Technique. *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, *3*(7), 10561–10568. https://doi.org/10.15662/ijareeie.2014.0307025

Prakash, R., & Singh, S. (2010). Improving RAID Performance and Reliability with Non-volatile Write Journals. *CYPress Perform*, (March), 1–5.

Purcaru, E., & Roma, C. (2011). 2D Barcode for DNA Encoding. *Journal of Mobile, Embedded and Distributed Systems*, *3*(3), 142–153. Retrieved from http://jmeds.eu/index.php/jmeds/article/view/2D-Barcode-for-DNA-Encoding/pdf

Qianyu, J. (2014). *Exploring the Concept of QR Code and the Benefits of Using QR Code for Companies*. Lapland University of Applied Sciences.

Rajasekar, S., Philominathan, P., & Chinnathambi, V. (2013). *Research methodology*. *Research Methodology*.

Ramya, M., & Jayasheela, M. (2014). Improved Color QR Codes for Real Time Applications with High Embedding Capacity. *International Journal of Computer Applications*, *91*(8), 8–12.

Rathod Rinkalkumar, M. (2014). A Review on 1D & 2D Barcode with QR Code Basic Structure and Characteristics. *International Journal of Futuristic Trends in Engineering and Technology Vol. 4 (01), 2014*, *4*(1), 4–7.

Rawat, D., Sahu, R., & Puthran, Y. (2015). Optimizing the Capacity of QR Code To Store Encrypted Image. *International Journal of Emerging Trends in Engineering Research (IJETER)*, *3*(1), 1–4.

Richard L. Villars, Olofson, C. W., & Eastwood, M. (2011). *Big Data: What it is and Why you should care. IDC*. MA, USA.

Rouse, M. (2015). Multiplexing. Retrieved July 28, 2018, from https://searchnetworking.techtarget.com/definition/multiplexing

Rungraungsilp, S., Ketcham, M., Wiputtikul, T., & Phonphak, K. (2012). Data Hiding Method for QR Code Based on Watermark by comparing DFT with DWT Domain. *International Conference on Computer and Communication Technologies (ICCCT'2012)*, *1*(1), 154–158.

Ryu, J. (2015). *The Modernization of the QR Code through Color and Brightness Level*. *John J. Ryu's Science Research*. Indiana. Retrieved from https://joonuryusciencerесearch.wordpress.com/2015/06/05/the-modernization-of-the-qr-code-through-color-and-brightness-levels/

Sangkwon, H., Hyung, J. B., Junhoi, K., Sunghwan, S., Sunghoon, K., & Wook. (2012). Drug Authentication Using High Capacity and Error Correctable Encoded Microtaggants. In *The 16th International Conference on Miniaturized Systems for Chemistry and Life Sciences* (pp. 1429–1431).

Sarkar, S., Pu, L., Wu, H. C., Huang, S. C. H., & Wu, Y. (2017). New multimedia archiving technique using multiple quick-response codes. In *IEEE International Symposium on Broadband Multimedia Systems and Broadcasting, BMSB* (pp. 4–9). https://doi.org/10.1109/BMSB.2017.7986236

Shahbahrami, A., Bahrampour, R., Rostami, M. S., & Mostafa Ayoubi. (2011). Evaluation of Huffman and Arithmetic Algorithms for Multimedia Compression Standards. *International Journal of Computer Science, Engineering and Applications*, *1*(4), 34–47. https://doi.org/10.5121/ijcsea.2011.1404

Sharma, M. (2010). Compression Using Huffman Coding. *IJCSNS International Journal of Computer Science and Network Security*, *10*(5, (May), 133–141.

Sharma, S., & Sejwar, V. (2016). QR code steganography for multiple image and text hiding using improved RSA-3DWT algorithm. *International Journal of Security and Its Applications*, *10*(7), 393–406. https://doi.org/10.14257/ijsia.2016.10.7.35

Shen, S., Lu, X., Qi, H., & Jiang, X. (2013). A Robust QR Code Extraction Algorithm. In Z. Wen & T. Li (Eds.), *Eighth International Conference on Intelligent Systems and Knowledge Engineering* (Vol. 2013, pp. 475–484). Springer Heidelberg New York Dordrecht London.

Shen, S., Lu, X., Qi, H., & Jiang, X. (2014). A Robust QR Code Extraction Algorithm. *Advances in Intelligent Systems and Computing*, *1*, 475–484. https://doi.org/10.1007/978-3-642-54924-3

Shiang-yen, T. A. N., Foo, L. Y., & Idrus, R. (2010). Application of Quick Response (QR) Codes in Mobile Tagging System for Retrieving Information about Genetically Modified Food. *Proceedings of the 9th WSEAS International Conference on Data Networks, Communications, Computers, DNCOCO*, (June 2015), 114–118. https://doi.org/978-960-474-245

Singh, A., Verma, V., & Raj, G. (2017). A Novel Approach for Encoding and Decoding of High Storage Capacity Color QR Code. In *2017 7th International Conference on Cloud Computing, Data Science & Engineering* (pp. 425–430).

Skawattananon, C., & Vongpradhip, S. (2013). An improved method to embed larger image in QR code. In *Proceedings of the 2013 10th International Joint Conference on Computer Science and Software Engineering, JCSSE 2013* (pp. 64–69). https://doi.org/10.1109/JCSSE.2013.6567321

Stinner, V. (2017). *Programming with Unicode Documentation Release 2011* (1st ed.). Sphinx. Retrieved from https://media.readthedocs.org/pdf/unicodebook/latest/unicodebook.pdf

Subpratatsavee, P., & Kuacharoen, P. (2012). An Implementation of a High Capacity 2D Barcode. *Advances in Information Technology Communications in Computer and Information Science*, *344*, 159–169.

Sun, M., Fang, Z., Fu, L., & Zhao, F. (2009). Identification of QR Codes Based on Pattern Recognition. In *Computer and Computing Technologies in Agriculture-Ⅱ--Proceedings of the Third IFIP International Conference on Computer and Computing Technologies in Agriculture(CCTA 2009)* (pp. 397–401).

Sundaram, V., Wood, T., & Prashant, S. (2006). Efficient Data Migration in Self-managing Storage Systems. In *Autonomic Computing, 2006. ICAC '06. IEEE International Conference* (p. 4).

Süsstrunk, S., Buckley, R., & Swen, S. (1999). Standard RGB Color Spaces. In *IS&T/SID 7th Color Imaging Conference* (pp. 1–8).

Sutheebanjard, P., & Premchaiswadi, W. (2010). QR Code Generator. In *Eighth International Conference on ICT and Knowledge Engineering* (pp. 89–92).

Szövetség, M. A., & Várallyai, L. (2012). From barcode to QR code applications. *Szám Journal of Agricultural Informatics*, *3*(2), 9–17. Retrieved from http://www.magisz.org/journal

Tank, A. H., Unde, M. M., Patel, B. J., & Raskar, P. (2016). Storage and transmission of information using grey level QR (quick-response) code structure. In *Conference on Advances in Signal Processing, CASP 2016* (pp. 402–405). https://doi.org/10.1109/CASP.2016.7746204

Taveerad, N., & Vongpradhip, S. (2016). Development of Color QR Code for Increasing Capacity. *Proceedings - 11th International Conference on Signal-Image Technology and Internet-Based Systems, SITIS 2015*, 645–648. https://doi.org/10.1109/SITIS.2015.42

Thomas, A., & Paul, R. (2013). An Effective Method for Removing Scratches and Restoring Low -Quality QR Code Images. *International Journal for Advance Research in Engineering and Technology Wind to Your Thoughts*, *1*(V), 5–9.

Tiwari, S. (2017). An introduction to QR code technology. In *15th International Conference on Information Technology, ICIT 2016* (Vol. 1, pp. 39–44). https://doi.org/10.1109/ICIT.2016.38

Toh, S. R., Goh, W., & Yeo, C. K. (2016). Data exchange via multiplexed color QR codes on mobile devices. In *Wireless Telecommunications Symposium* (Vol. 2016–May). https://doi.org/10.1109/WTS.2016.7482035

Trochim, W. M. K. (2015). Research Methods: Knowledge base, *1*(1), 1–369. Retrieved from https://www.researchgate.net/publication/243783609_The_Research_Methods_Knowledge_Base

Umaria, M. M., & Jethava, G. B. (2015). Enhancing Data Storage Capacity in Quick Response Code Using Multiplexing and Data Compression Technique. *2015 International Conference on Computational Intelligence and Communication Networks (CICN)*, (1), 1091–1093. https://doi.org/10.1109/CICN.2015.214

Victor, N. (2012). Enhancing the Data Capacity of QR Codes by Compressing the Data before Generation. *International Journal of Computer Applications (0975 –8887)*, *60*(2), 17–21.

Vizcarra Melgar, M. E., Zaghetto, A., Macchiavello, B., & Nascimento, A. C. A. (2012). CQR codes: Colored quick-response codes. In *IEEE International Conference on Consumer Electronics - Berlin, ICCE-Berlin* (Vol. 2401, pp. 321–325). https://doi.org/10.1109/ICCE-Berlin.2012.6336526

Vongpradhip, S. (2013). Use Multiplexing to Increase Information in QR Code. In *The 8th International Conference on Computer Science & Education (ICCSE 2013)* (pp. 361–364).

Wakahara, T., Yamamoto, N., & Ochi, H. (2010). Image processing of dotted picture in the QR code of cellular phone. In *Proceedings - International Conference on P2P, Parallel, Grid, Cloud and Internet Computing, 3PGCIC 2010* (pp. 454–458). https://doi.org/10.1109/3PGCIC.2010.77

Wang, S., Yang, T., Li, J., Yao, B., & Zhang, Y. (2015). Does a QR code must be black and white? In *Proceedings of 2015 International Conference on Orange Technologies, ICOT 2015* (pp. 161–164). https://doi.org/10.1109/ICOT.2015.7498513

Warasart, M., & Kuacharoen, P. (2012). Paper-based Document Authentication using Digital Signature and QR Code. In *4th International Conference on Computer Engineering and Technology (ICCET 2012)*.

Weeks, H. R. M., Arthur, R., & Sartor, L. J. (1999). Histogram specification of 24-bit color images in the color difference (C-Y) color space. *Journal of Electronic Imaging*, *8*(3), 290–300.

Winter, M. (2011). *Scan Me - Everybody's Guide to the Magical World of Qr Codes* (1st ed.). California: Westsong Publishing. Retrieved from https://books.google.com/books?id=s5ZxqwwYKk8C&pgis=1

Yadav, S. H., & Dawande, P. N. A. (2016). A Survey on Image Embedding In QR Code. *International Journal of Advanced Research in Computer and Communication Engineering*, *5*(4), 339–341. https://doi.org/10.17148/IJARCCE.2016.5486

Yang, Z., Cheng, Z., Loy, C. C., Lau, W. C., Li, C. M., & Li, G. (2016). Towards Robust Color Recovery For High Capacity Color QR Codes. In *2016 IEEE International Conference on Image Processing (ICIP)* (pp. 2866–2870).

Yang, Z., Xu, H., Deng, J., Loy, C. C., & Lau, W. C. (2017). Robust and Fast Decoding of High-Capacity Color QR Codes for Mobile Applications. *IEEE Transactions on Image Processing*, *PP*(c), 1. https://doi.org/10.1109/TIP.2018.2855419

Yeap, K. H., Cheong, Y. K., Nisar, H., & Teh, P. C. (2014). A simple data storage system using QR code. In *2014 5th International Conference on Intelligent and Advanced Systems (ICIAS)* (pp. 1–5). https://doi.org/10.1109/ICIAS.2014.6869536

Zhang, M., Yao, D., & Zhou, Q. (2012). The Application and Design of QR Code in Scenic Spot's eTicketing System -A Case Study of Shenzhen Happy Valley. *International Journal of Science and Technology*, *2*(12). Retrieved from http://www.ejournalofsciences.org

Zhang, W., & Yang, T. (2015). An Improved Algorithm for QR Code Image Binarization. In *2014 International Conference on Virtual Reality and Visualization And Visualization* (pp. 154–159). https://doi.org/10.1109/ICVRV.2014.51

Zhang, Y., Gao, T., Li, D., & Lin, H. (2012). An Improved Binarization Algorithm of QR Code Image, (1), 2376–2379.

Zhu, B., Li, K., & Patterson, H. (2008). Avoiding the Disk Bottleneck in the Data Domain Deduplication File System Challenges and Observations. In *FAST '08: 6th USENIX Conference on File and Storage Technologies* (pp. 269–282).

Appendix A : Result of Maximum Characters

Result of maximum total characters stored in QR code from 20 times tested at error correction level H.

No. of Test	Normal	Zip	GZip	LZW	Huffmann Coding	Huffman+ GZip
1	1271	474	635	434	113	471
2	1271	471	638	434	112	466
3	1271	476	637	433	111	477
4	1271	472	636	436	114	474
5	1271	475	637	433	112	470
6	1271	473	635	438	112	473
7	1271	475	635	433	111	474
8	1271	473	641	438	111	472
9	1271	474	636	438	114	468
10	1271	474	638	439	113	470
11	1271	473	634	433	113	468
12	1271	473	637	438	111	474
13	1271	471	633	441	111	477
14	1271	471	634	433	111	472
15	1271	473	635	437	113	471
16	1271	469	636	440	111	471
17	1271	470	636	438	111	479
18	1271	469	633	437	113	471
19	1271	477	636	436	112	467
20	1271	478	632	433	109	467

Appendix B : Encode Level L

The algorithm of encoding process for error correction level L.

```
/* Algorithm Module Index E1 – Initialisation */
/*-------------------------------------------------*/

/* Creating the package to be used */
package qrcodecmm;

/* Initialisation java library */
call import java.io.FileInputStream;
call import java.io.FileOutputStream;
call import it.sauronsoftware.base64.Base64;
call import java.io.IOException;

/* Creating main class */
public class QRCodeCMM {

/* Initialisation variables to be used */
initialize static String plainTextFile = "D:/QR Code/QRCode/Journal3/levelL.txt";
initialize static String gZipTextFile = "D:/QR Code/QRCode/Journal3/gZipTextFile.gzip";
initialize static String base91TextFile = "D:/QR
Code/QRCode/Journal3/base91TextFile.b91";
initialize static String filePath = "D:/QR Code/QRCode/Journal3/";
initialize static String fileName = "fileNumber";
initialize static String fileType = ".txt";
initialize static String fileTypePNG = "png";
initialize static String fileRGB = filePath + "fileRGB." + fileTypePNG;
initialize static final int size = 551;

/* Creating main programme */
public static void main(String[] args) {

/* Counting the total characters including line feed and carriage return */
initialize object CounterLetters count = new CounterLetters();
execute count.CountLetter(plainTextFile);

/* Algorithm Module Index E2 – Compression utility (GZip) */
/*-------------------------------------------------*/

initialize object GZip gZip = new GZip();
execute gZip.gZipFile(plainTextFile, gZipTextFile);

/* Algorithm Module Index E3 – Encoder for Base64*/
/*-----------------------------------------------------------*/
```

```
initialize object base91cli base91 = new base91cli();
try {
initialize object FileInputStream ifs = new FileInputStream(gZipTextFile);
initialize object FileOutputStream ofs = new FileOutputStream(base91TextFile);
execute base91.encode(ifs, ofs);
} catch (Exception e) {
display error by using System.err.println(e);
}

/* Algorithm Module Index E4 – Compatibility QR Code ANSI to UTF */
/*----------------------------------------------------------------*/

initialize object ansiToUTF8 utf8 = new ansiToUTF8();
execute utf8.convert(base91TextFile);

/* Algorithm Module Index E5 – Creating blank file */
/*----------------------------------------------------------------*/

initialize object CreateQRCode1 create1 = new CreateQRCode1(size);
initialize object CreateQRCode9 create9 = new CreateQRCode9(size);
initialize object CreateQRCode17 create17 = new CreateQRCode17(size);

/* Counting characters */
initialize object  CountChar countChar = new CountChar();
execute int countCharacter = countChar.count(base91TextFile);

/* Divide characters with related value and fit each of file */
initialize object  DivideCharacters divide = new DivideCharacters();

/* Create blank files */
execute create1.createBlank40Files(filePath, fileName, fileType);


/* Algorithm Module Index E6 – Embedded text characters to each file */
/*----------------------------------------------------------------------------------*/

/* Embedded text characters  to each blank file created*/
execute int totalFiles = divide.divideCharacter(base91TextFile, countCharacter, filePath,
fileName);

/* Top up with blank files if not enough */
      if (totalFiles < 25) {
         totalFiles = 24;
      }

/* Specify three group contains 8 files each */
initialize int eight = 8;
initialize String[] multiColourLayerFail = {"QRRed", "QRGreen", "QRBlue"};
initialize int colourCombineRGB[][][] = new int[size][size][3];
initialize MultiLayerQRCode multiLayerQRCode = new MultiLayerQRCode();
```

```
/* Algorithm Module Index E7 and E8– Create black and white QR Code and
monocoloured QR Code.  */
/*-------------------------------------------------------------------------------------------------
-*/

/* Start with first  group (red) */
if (totalFiles >= 0) {
        initialize int total8 = 8;

        /* Create black and white QR Code from 1 to 8 (Module Index E7)
        execute create1.generateQRCodeVersion40(filePath, fileName, fileType,
fileTypePNG, total8);

        /* Create monocoloured QR Code group 1 (Module Index E8)
        try {
           execute resultFinal = create1.readImage(filePath, fileName, fileTypePNG, total8);
        } catch (IOException ex) {
        }
        execute int[][] plotResultBlackWhite =
create1.generateMultiplexQRCode(resultFinal, total8);

        /* Combine pixels among 8 black and white QR Codes */
        for (int x = 0; x < size; x++) {
           for (int y = 0; y < size; y++) {
             execute colourCombineRGB[x][y][0] = plotResultBlackWhite[x][y];
           }
        }

        /* Generate first monocoloured QR Code */
        execute create1.generateQRCodeVersion40MonoColour(filePath, fileTypePNG,
plotResultBlackWhite, multiColourLayerFail[0]);
}

/* Start with second  group (green) */
  if (totalFiles >= 8) {
        initialize int total16 = 16;

        /* Create black and white QR Code from 9 to 16 (Module Index E7)
        create9.generateQRCodeVersion40(filePath, fileName, fileType, fileTypePNG,
total16);
        initialize int[][][] resultFinal = new int[eight][size][size];

        /* Create monocoloured QR Code group 2 (Module Index E8)
        try {
           execute resultFinal = create9.readImage(filePath, fileName, fileTypePNG, total16,
eight);
        } catch (IOException ex) {
        }
```

```
        execute  int[][] plotResultBlackWhite =
create9.generateMultiplexQRCode(resultFinal, total16);

        /* Combine pixels among 8 black and white QR Codes */
        for (int x = 0; x < size; x++) {
           for (int y = 0; y < size; y++) {
              execute colourCombineRGB[x][y][1] = plotResultBlackWhite[x][y];
           }
        }

        /* Generate second monocoloured QR Code */
        execute create9.generateQRCodeVersion40MonoColour(filePath, fileTypePNG,
plotResultBlackWhite, multiColourLayerFail[1]);
        }

      /* Start with third group (blue) */
      if (totalFiles >= 16) {
        initialize int total24 = 24;

        /* Create black and white QR Code from 17 to 24 (Module Index E7)
        execute create17.generateQRCodeVersion40(filePath, fileName, fileType,
fileTypePNG, total24);
        initialize int[][][] resultFinal = new int[eight][size][size];

        /* Create monocoloured QR Code group 3 (Module Index E8)
        try {
           execute resultFinal = create17.readImage(filePath, fileName, fileTypePNG,
total24, eight);
        } catch (IOException ex) {
        }

        execute int[][] plotResultBlackWhite =
create17.generateMultiplexQRCode(resultFinal, total24);

        /* Combine pixels among 8 black and white QR Codes */
        for (int x = 0; x < size; x++) {
           for (int y = 0; y < size; y++) {
              execute colourCombineRGB[x][y][2] = plotResultBlackWhite[x][y];
           }
        }

        /* Generate third monocoloured QR Code */
       execute create17.generateQRCodeVersion40MonoColour(filePath, fileTypePNG,
plotResultBlackWhite, multiColourLayerFail[2]);
      }

/* Algorithm Module Index E9– Create coloured QR Code.  */
/*-----------------------------------------------------------------------------------------------------
-*/
```

```
        initialize CombineRGB combineRGBColour = new CombineRGB();
        execute combineRGBColour.combineRGB(colourCombineRGB, fileRGB);
    }
}
```

Appendix C : Decode Level L

The algorithm of decoding process for error correction level L.

```
/* Algorithm Module Index D1 – Initialisation */
/*-------------------------------------------------*/

/* Creating the package to be used */
package qrcodecmn;

/* Initialisation variables to be used */

call import com.google.zxing.NotFoundException;
call import com.google.zxing.WriterException;
call import java.io.FileInputStream;
call import java.io.FileOutputStream;
call import it.sauronsoftware.base64.Base64;
call import java.awt.Colour;
call import java.io.BufferedWriter;
call import java.io.File;
call import java.io.FileWriter;
call import java.io.IOException;

/* Creating main class */
public class QRCodeCMN {

/* Initialisation variables to be used */
initialize static String plainTextFile = "D:/QR Code/QRCode/Journal3/Decode/fileText.txt";
initialize static String gZipTextFile = "D:/QR
Code/QRCode/Journal3/Decode/gZipTextFile.gzip";
initialize static String base91TextFile = "D:/QR
Code/QRCode/Journal3/Decode/base91TextFile.b91";
initialize static String filePath = "D:/QR Code/QRCode/Journal3/Decode/";
initialize static String filePathBefore = "D:/QR Code/QRCode/Journal3/";
initialize static String fileName = "fileNumberMerged";
initialize static String fileType = ".txt";
initialize static String fileTypePNG = "png";
initialize static String fileRGB = filePath + "fileRGB." + fileTypePNG;
initialize static String fileRGBBefore = filePathBefore + "fileRGB." + fileTypePNG;
initialize static File fileRGBDecode = new File(fileRGBBefore);
initialize static String[] multiColourLayerFile = {"QRRedDecode", "QRGreenDecode",
“QRBlueDecode"};
initialize static String fileRed = filePath + multiColourLayerFile[0] + "." + fileTypePNG;
initialize static String fileGreen = filePath + multiColourLayerFile[1] + "." + fileTypePNG;
initialize static String fileBlue = filePath + multiColourLayerFile[2] + "." + fileTypePNG;
initialize static File fileRedDecode = new File(fileRed);
initialize static File fileGreenDecode = new File(fileGreen);
```

initialize static File fileBlueDecode = new File(fileBlue);
initialize static String fileQRCodeBlackWhite[] = {"QRCodeBlackWhiteRed",
“QRCodeBlackWhiteGreen", "QRCodeBlackWhiteBlue"};
initialize static String fileOutputTextDecode = filePath + fileName + fileType;
initialize static String plainTextFileDecode = "D:/QR
Code/QRCode/Journal3/Decode/plainTextFile.txt";
initialize static String gZipTextFileDecode = "D:/QR
Code/QRCode/Journal3/Decode/gZipTextFile.gzip";
initialize static String base91TextFileDecode = "D:/QR
Code/QRCode/Journal3/Decode/fileNumberMerged.txt";
initialize static long startdecodeMultilayer, stopdecodeMultilayer;
initialize static long startdecodeRedQRCode, startdecodeGreenQRCode,
startdecodeBlueQRCode;
initialize static long stopdecodeRedQRCode, stopdecodeGreenQRCode,
stopdecodeBlueQRCode;
initialize static long startdecodeRedBlackQRCode, startdecodeGreenBlackQRCode,
startdecodeBlueBlackQRCode;
initialize static long stopdecodeRedBlackQRCode, stopdecodeGreenBlackQRCode,
stopdecodeBlueBlackQRCode;
initialize static long startdecodeBlackQRCode, stopdecodeBlackQRCode;
initialize static long startdecodebase91, stopdecodebase91;
initialize static long startdecodeGUnzip, stopdecodeGUnzip;
initialize static long startdecodeAll, stopdecodeAll;

/* Creating main class */
public static void main(String[] args) {

initialize object DecodeColourQR decode = new DecodeColourQR();
initialize object DecodeQRCode QRCodeText = new DecodeQRCode();
initialize int size = 551;
initialize int files = 8;

/* Algorithm Module Index D2 – Demultilayer*/
/*----------*/

/* Decode From Coloured To Red, Green, and Blue Monocoloured */
try {
	execute Colour[][] resultcolourQRCodeDecode =
decode.readImage(fileRGBDecode);
	execute decode.decodeMultiLayerQRCodeRGB1(resultcolourQRCodeDecode,
filePath, multiColourLayerFile);

/* Algorithm Module Index D3 – Demultiplexing*/
/*----------*/

/* Initialize the information of Black and White QR Code */
initialize QRCodeBlackWhite = new int[files][size][size];

/* Demultiplexing Red QR Code */

```
execute Colour[][] resultcolourRedQRCodeDecode = decode.readImage(fileRedDecode);
execute QRCodeBlackWhite =
decode.decodeQRCodeBlackWhite(resultcolourRedQRCodeDecode, 0);
execute decode.decodeQRCodeBlackWhite1(QRCodeBlackWhite, filePath,
fileQRCodeBlackWhite[0]);

/* Demultiplexing Green QR Code */
execute Colour[][] resultcolourGreenQRCodeDecode =
decode.readImage(fileGreenDecode);
execute QRCodeBlackWhite =
decode.decodeQRCodeBlackWhite(resultcolourGreenQRCodeDecode, 1);
execute decode.decodeQRCodeBlackWhite1(QRCodeBlackWhite, filePath,
fileQRCodeBlackWhite[1]);

/* Demultiplexing Blue QR Code */
execute Colour[][] resultcolourBlueQRCodeDecode = decode.readImage(fileBlueDecode);
execute QRCodeBlackWhite =
decode.decodeQRCodeBlackWhite(resultcolourBlueQRCodeDecode, 2);
execute decode.decodeQRCodeBlackWhite1(QRCodeBlackWhite, filePath,
fileQRCodeBlackWhite[2]);
} catch (IOException ex) {
}

/* Algorithm Module Index D4 – Decode Black and White QR Code */
/*-------------------------------------------------------------------------*/

/* Counting the total characters including line feed and carriage return */
initialize object CounterLetters count = new CounterLetters();
try {
initialize object BufferedWriter writer = null;
initialize and excute writer = new BufferedWriter(new FileWriter(fileOutputTextDecode));

/* Creating Naming Conversion for Black and White QR Code Image Name */
for (int i = 0; i < 24; i++) {
     initialize String filePathDecodeFinal = null;
          if ((i >= 0) && (i < 8)) {
            initialize filePathDecodeFinal = fileQRCodeBlackWhite[0] + i + "." +
fileTypePNG;

          } else if ((i >= 8) && (i < 16)) {
             initialize filePathDecodeFinal = fileQRCodeBlackWhite[1] + (i - 8) + "." +
fileTypePNG;

          } else if ((i >= 16) && (i < 24)) {
             initialize filePathDecodeFinal = fileQRCodeBlackWhite[2] + (i - 16) + "." +
fileTypePNG;
          }

initialize String filePathDecodeComplete = filePath + filePathDecodeFinal;
```

```
/* Decode Black and White QR Code */
initialize and excute String valueText =
QRCodeText.decodeTextQRCode(filePathDecodeComplete);

/* Write to Text File */
execute writer.write(valueText);

 }
        writer.close();
 } catch (WriterException ex) {
        ex.printStackTrace();
 } catch (IOException ey) {
        ey.printStackTrace();
 } catch (NotFoundException ez) {
        ez.printStackTrace();
 }

/* Algorithm Module Index D5 – Decode Encoder/Decoder Text (Base64) to
Compression file */
/*----------------------------------------------------------------------------------------------------*/

initialize Base64cli base64 = new base64cli();

try {
        initialize object FileInputStream ifs = new FileInputStream(base91TextFileDecode);
        initialize object FileOutputStream ofs = new FileOutputStream(gZipTextFileDecode);

        execute Base64.decode(ifs, ofs);
} catch (Exception e) {
System.err.println(e);
}

/* Algorithm Module Index D6 –Extraction Compression File to Text File */
/*----------------------------------------------------------------------------------------------------*/
 initialize object GZip gZip = new GZip();
 execute gZip.gunzipIt(gZipTextFileDecode, plainTextFileDecode);
}
}
```

Appendix D : Partial Extraction (Decode) Level 1

The algorithm of partial extraction level 1 (decode) process for error correction level L.

```
/* Algorithm Module Index P1 – Initialisation */
/*-------------------------------------------------*/

/* Creating the package to be used */
package qrcodecmn;

/* Initialisation java library */
call import com.google.zxing.NotFoundException;
call import com.google.zxing.WriterException;
call import java.io.FileInputStream;
call import java.io.FileOutputStream;
call import it.sauronsoftware.base64.Base64;
call import java.awt.Colour;
call import java.io.BufferedWriter;
call import java.io.File;
call import java.io.FileWriter;
call import java.io.IOException;

/* Creating main class */
public class QRCodeCMN {

initialize static String plainTextFile = "D:/QR Code/QRCode/Journal3/Decode/fileText.txt";
initialize static String gZipTextFile = "D:/QR
Code/QRCode/Journal3/Decode/gZipTextFile.gzip";
initialize static String base91TextFile = "D:/QR
Code/QRCode/Journal3/Decode/base91TextFile.b91";
initialize static String filePath = "D:/QR Code/QRCode/Journal3/Decode/";
initialize static String filePathBefore = "D:/QR Code/QRCode/Journal3/";
initialize static String fileName = "fileNumberMerged";
initialize static String fileType = ".txt";
initialize static String fileTypePNG = "png";
initialize static String fileRGB = filePath + "fileRGB." + fileTypePNG;
initialize static String fileRGBBefore = filePathBefore + "fileRGB." + fileTypePNG;
initialize static File fileRGBDecode = new File(fileRGBBefore);
initialize static String[] multiColourLayerFile = {"QRRedDecode", "QRGreenDecode",
"QRBlueDecode"};
initialize static String fileRed = filePath + multiColourLayerFile[0] + "." + fileTypePNG;
initialize static String fileGreen = filePath + multiColourLayerFile[1] + "." + fileTypePNG;
initialize static String fileBlue = filePath + multiColourLayerFile[2] + "." + fileTypePNG;
initialize static File fileRedDecode = new File(fileRed);
initialize static File fileGreenDecode = new File(fileGreen);
initialize static File fileBlueDecode = new File(fileBlue);
```

```
initialize static String fileQRCodeBlackWhite[] = {"QRCodeBlackWhiteRed",
"QRCodeBlackWhiteGreen", "QRCodeBlackWhiteBlue"};
initialize static String fileOutputTextDecode = filePath + fileName + fileType;
initialize static String plainTextFileDecode = "D:/QR
Code/QRCode/Journal3/Decode/plainTextFile.txt";
initialize static String gZipTextFileDecode = "D:/QR
Code/QRCode/Journal3/Decode/gZipTextFile.gzip";
initialize static String base91TextFileDecode = "D:/QR
Code/QRCode/Journal3/Decode/fileNumberMerged.txt";
initialize static int blackWhiteQRCode = 0;


/* Creating main programme */
public static void main(String[] args) {

initialize object DecodeColourQR decode = new DecodeColourQR();
initialize object DecodeQRCode QRCodeText = new DecodeQRCode();
initialize int size = 551;
initialize int files = 8;

/* Algorithm Module Index P2 – Extracting coloured QR Code  (Demultilayer) */
/*-------------------------------------------------*/

try {

execute Colour[][] resultcolourQRCodeDecode = decode.readImage(fileRGBDecode);
execute decode.decodeMultiLayerQRCodeRGB1(resultcolourQRCodeDecode, filePath,
multiColourLayerFile);

/* Algorithm Module Index D3 – Demultiplexing*/
/*-----------------------------------------------------*/

/* Initialize the information of Black and White QR Code */
initialize QRCodeBlackWhite = new int[files][size][size];

/* Demultiplexing Red QR Code */
execute Colour[][] resultcolourRedQRCodeDecode = decode.readImage(fileRedDecode);
execute QRCodeBlackWhite =
decode.decodeQRCodeBlackWhite(resultcolourRedQRCodeDecode, 0);
execute decode.decodeQRCodeBlackWhite1(QRCodeBlackWhite, filePath,
fileQRCodeBlackWhite[0]);
 } catch (IOException ex) {
 }

/* Algorithm Module Index P4 – Decode Black and White QR Code */
/*-------------------------------------------------------------------------*/
/* Counting the total characters including line feed and carriage return */
initialize object CounterLetters count = new CounterLetters();

try {
```

```
initialize object BufferedWriter writer = null;
initialize and excute writer = new BufferedWriter(new FileWriter(fileOutputTextDecode));

/* Creating Single Naming Conversion for a Black and White QR Code Image Name */

initialize String filePathDecodeFinal = null;
execute int i = blackWhiteQRCode;
initialize filePathDecodeFinal = fileQRCodeBlackWhite[0] + i + "." + fileTypePNG;

initialize String filePathDecodeComplete = filePath + filePathDecodeFinal;

/* Decode Single Black and White QR Code */
initialize and execute String valueText =
QRCodeText.decodeTextQRCode(filePathDecodeComplete);

/* Write to Text File */
execute writer.write(valueText);

/* Close to Text File */
execute writer.close();

} catch (WriterException ex) {
        ex.printStackTrace();
 } catch (IOException ey) {
        ey.printStackTrace();
 } catch (NotFoundException ez) {
        ez.printStackTrace();
 }

/* Algorithm Module Index P5 – Decode Encoder/Decoder Text (Base64) to
Compression file */
/*-------------------------------------------------------------------------------------------------------*/

initialize Base64cli base64 = new base64cli();

try {
        initialize object FileInputStream ifs = new FileInputStream(base91TextFileDecode);
        initialize object FileOutputStream ofs = new FileOutputStream(gZipTextFileDecode);

        execute Base64.decode(ifs, ofs);
} catch (Exception e) {
System.err.println(e);
}
/* Algorithm Module Index P6 –Extraction Compression File to Text File */
/*-------------------------------------------------------------------------------------------------------*/
 initialize object GZip gZip = new GZip();
 execute gZip.gunzipIt(gZipTextFileDecode, plainTextFileDecode);

/* Algorithm Module Index P7 – Open  Text Application and Ready to Manipulate*/
/*-------------------------------------------------------------------------------------------------------*/
```

execute “C:\windows\systems\notepad.exe”
}
}

Appendix E : Partial Extraction (Re-encode) Level 1

The algorithm of partial extraction level 1 (re-encode) process for error correction level L.

```
/* Algorithm Module Index P8 – Initialisation */
/*----------------------------------------------------*/

/* Creating the package to be used */
package qrcodecmm;

/* Initialisation java library */
call import java.io.FileInputStream;
call import java.io.FileOutputStream;
call import it.sauronsoftware.base64.Base64;
call import java.io.IOException;

/* Creating main class */
public class QRCodeCMM {

/* Initialisation variables to be used */
initialize static String plainTextFile = "D:/QR
Code/QRCode/Journal3/singleTextLevel1L.txt";
initialize static String gZipTextFile = "D:/QR Code/QRCode/Journal3/gZipTextFile.gzip";
initialize static String base64TextFile = "D:/QR
Code/QRCode/Journal3/base64TextFile.b64";
initialize static String filePath = "D:/QR Code/QRCode/Journal3/";
initialize static String fileName = "fileNumber";
initialize static String fileType = ".txt";
initialize static String fileTypePNG = "png";
initialize static String fileRGB = filePath + "fileRGB." + fileTypePNG;
initialize static final int size = 551;

/* Creating main programme */
public static void main(String[] args) {

/* Counting the total characters including line feed and carriage return */
initialize object CounterLetters count = new CounterLetters();
execute count.CountLetter(plainTextFile);

/* Algorithm Module Index P9 – Compression utility (GZip) */
/*----------------------------------------------------*/

initialize object GZip gZip = new GZip();
execute gZip.gZipFile(plainTextFile, gZipTextFile);

/* Algorithm Module Index P10 – Encoder for Base64*/
/*-------------------------------------------------------------*/
```

```
initialize object base91cli base91 = new base91cli();
try {
initialize object FileInputStream ifs = new FileInputStream(gZipTextFile);
initialize object FileOutputStream ofs = new FileOutputStream(base64TextFile);
execute base91.encode(ifs, ofs);
} catch (Exception e) {
display error System.err.println(e);
}

/* Algorithm Module Index P11 – Compatibility QR Code ANSI to UTF */
/*----------------------------------------------------------------*/

initialize object ansiToUTF8 utf8 = new ansiToUTF8();
execute utf8.convert(base64TextFile);

/* Algorithm Module Index P12 – Creating Black and White QR Code */
/*----------------------------------------------------------------*/

initialize object CreateQRCode1 create1 = new CreateQRCode1(size);
initialize object CreateQRCode9 create9 = new CreateQRCode9(size);
initialize object CreateQRCode17 create17 = new CreateQRCode17(size);

/* Counting characters */
initialize object  CountChar countChar = new CountChar();
execute int countCharacter = countChar.count(base64TextFile);

/* Divide characters with related value and fit each of file */
initialize object  DivideCharacters divide = new DivideCharacters();

/* Create blank files */
execute create1.createBlank40Files(filePath, fileName, fileType);

/* Embedded text characters  to each blank file created*/
execute int totalFiles = divide.divideCharacter(base64TextFile, countCharacter, filePath,
fileName);

/* Checking if the file exceed more than 1 */
      if (totalFiles >= 2) {
         System.exit();
      }

/* Specify three group that contains 8 files each */
initialize int eight = 8;
initialize String[] multiColourLayerFail = {"QRRed", "QRGreen", "QRBlue"};
initialize int colourCombineRGB[][][] = new int[size][size][3];
initialize MultiLayerQRCode multiLayerQRCode = new MultiLayerQRCode();

/* Algorithm Module Index P12 and P13– Create black and white QR Code and place it
in a group.  */
```

```
/*-------------------------------------------------------------------------------------------------------
-*/

/* Start with first  group (red) */
if (totalFiles >= 0) {
        initialize int total8 = 8;

        /* Create an updated single black and white QR Code (Module Index P12)
        execute create1.generateQRCodeVersion40(filePath, fileName, fileType,
fileTypePNG, total8);

        /* get information from QR Code group 1 (Module Index P13)
        try {
           execute resultFinal = create1.readImage(filePath, fileName, fileTypePNG, total8);
        } catch (IOException ex) {
        }
        execute int[][] plotResultBlackWhite =
create1.generateMultiplexQRCode(resultFinal, total8);

        /* Combine pixels among 8 black and white QR Codes */
        for (int x = 0; x < size; x++) {
           for (int y = 0; y < size; y++) {
             execute colourCombineRGB[x][y][0] = plotResultBlackWhite[x][y];
           }
        }

/* Algorithm Module Index P14 – Create monocoloured QR Code.  */
/*-------------------------------------------------------------------------------------------------------
-*/
        execute create1.generateQRCodeVersion40MonoColour(filePath, fileTypePNG,
plotResultBlackWhite, multiColourLayerFail[0]);
}
      }

/* Algorithm Module Index P15– Create coloured QR Code.  */
/*-------------------------------------------------------------------------------------------------------
-*/

      initialize CombineRGB combineRGBColour = new CombineRGB();
      execute combineRGBColour.combineRGB(colourCombineRGB, fileRGB);
   }
}
```

Appendix F : Partial Extraction (Decode) Level 2

The algorithm of partial extraction level 2 (decode) process for error correction level L.

```
/* Algorithm Module Index P16 – Initialisation */
/*----------------------------------------------------*/

/* Creating the package to be used */
package qrcodecmn;

/* Initialisation variables to be used */

call import com.google.zxing.NotFoundException;
call import com.google.zxing.WriterException;
call import java.io.FileInputStream;
call import java.io.FileOutputStream;
call import it.sauronsoftware.base64.Base64;
call import java.awt.Colour;
call import java.io.BufferedWriter;
call import java.io.File;
call import java.io.FileWriter;
call import java.io.IOException;

/* Creating main class */
public class QRCodeCMN {

/* Initialisation variables to be used */
initialize static String plainTextFile = "D:/QR Code/QRCode/Journal3/Decode/
singleTextLevel2L.txt ";
initialize static String gZipTextFile = "D:/QR
Code/QRCode/Journal3/Decode/gZipTextFile.gzip";
initialize static String base91TextFile = "D:/QR
Code/QRCode/Journal3/Decode/base91TextFile.b91";
initialize static String filePath = "D:/QR Code/QRCode/Journal3/Decode/";
initialize static String filePathBefore = "D:/QR Code/QRCode/Journal3/";
initialize static String fileName = "fileNumberMerged";
initialize static String fileType = ".txt";
initialize static String fileTypePNG = "png";
initialize static String fileRGB = filePath + "fileRGB." + fileTypePNG;
initialize static String fileRGBBefore = filePathBefore + "fileRGB." + fileTypePNG;
initialize static File fileRGBDecode = new File(fileRGBBefore);
initialize static String[] multiColourLayerFile = {"QRRedDecode", "QRGreenDecode",
“QRBlueDecode"};
initialize static String fileRed = filePath + multiColourLayerFile[0] + "." + fileTypePNG;
initialize static String fileGreen = filePath + multiColourLayerFile[1] + "." + fileTypePNG;
initialize static String fileBlue = filePath + multiColourLayerFile[2] + "." + fileTypePNG;
initialize static File fileRedDecode = new File(fileRed);
```

```
initialize static File fileGreenDecode = new File(fileGreen);
initialize static File fileBlueDecode = new File(fileBlue);
initialize static String fileQRCodeBlackWhite[] = {"QRCodeBlackWhiteRed",
“QRCodeBlackWhiteGreen", "QRCodeBlackWhiteBlue"};
initialize static String fileOutputTextDecode = filePath + fileName + fileType;
initialize static String plainTextFileDecode = "D:/QR
Code/QRCode/Journal3/Decode/plainTextFile.txt";
initialize static String gZipTextFileDecode = "D:/QR
Code/QRCode/Journal3/Decode/gZipTextFile.gzip";
initialize static String base91TextFileDecode = "D:/QR
Code/QRCode/Journal3/Decode/fileNumberMerged.txt";

/* Creating main class */
public static void main(String[] args) {

initialize object DecodeColourQR decode = new DecodeColourQR();
initialize object DecodeQRCode QRCodeText = new DecodeQRCode();
initialize int size = 551;
initialize int files = 8;

/* Algorithm Module Index P17 – Demultilayer*/
/*----------------------------------------------------------*/

/* Decode From Coloured To Red, Green, and Blue Monocoloured */
try {
        execute Colour[][] resultcoluorQRCodeDecode =
decode.readImage(fileRGBDecode);
        execute decode.decodeMultiLayerQRCodeRGB1(resultcolourQRCodeDecode,
filePath, multiColourLayerFile);

/* Algorithm Module Index P18 – Demultiplexing*/
/*----------------------------------------------------------*/

/* Initialize the information of Black and White QR Code */
initialize QRCodeBlackWhite = new int[files][size][size];

/* Demultiplexing Red QR Code */
execute Colour[][] resultcolourRedQRCodeDecode = decode.readImage(fileRedDecode);
execute QRCodeBlackWhite =
decode.decodeQRCodeBlackWhite(resultcolourRedQRCodeDecode, 0);
execute decode.decodeQRCodeBlackWhite1(QRCodeBlackWhite, filePath,
fileQRCodeBlackWhite[0]);

/* Algorithm Module Index P19 – Decode Selected Group of Black and White QR Code
*/
/*----------------------------------------------------------------------------*/

/* Counting the total characters including line feed and carriage return */
initialize object CounterLetters count = new CounterLetters();
try {
```

```
initialize object BufferedWriter writer = null;
initialize and excute writer = new BufferedWriter(new FileWriter(fileOutputTextDecode));

/* Identify Index Location and Creating Naming Conversion of a Group  Black and
White QR Code Image Names */
for (int i = 0; i < 8; i++) {
     initialize String filePathDecodeFinal = null;
          if ((i >= 0) && (i < 8)) {
            initialize filePathDecodeFinal = fileQRCodeBlackWhite[0] + i + "." +
fileTypePNG;
}

initialize String filePathDecodeComplete = filePath + filePathDecodeFinal;

/* Decode Black and White QR Code */
initialize and excute String valueText =
QRCodeText.decodeTextQRCode(filePathDecodeComplete);

/* Write to Text File */
execute writer.write(valueText);

 }
        writer.close();
 } catch (WriterException ex) {
        ex.printStackTrace();
 } catch (IOException ey) {
        ey.printStackTrace();
 } catch (NotFoundException ez) {
        ez.printStackTrace();
 }

/* Algorithm Module Index P20 – Decode Encoder/Decoder Text (Base64) to
Compression file */
/*----------------------------------------------------------------------------------------------------*/

initialize Base64cli base64 = new base64cli();

try {
        initialize object FileInputStream ifs = new FileInputStream(base91TextFileDecode);
        initialize object FileOutputStream ofs = new FileOutputStream(gZipTextFileDecode);

        execute Base64.decode(ifs, ofs);
} catch (Exception e) {
System.err.println(e);
}

/* Algorithm Module Index P21 –Extraction Compression File to Text File */
/*----------------------------------------------------------------------------------------------------*/
 initialize object GZip gZip = new GZip();
 execute gZip.gunzipIt(gZipTextFileDecode, plainTextFileDecode);
```

/* Algorithm Module Index P22 – Open Text Application and Ready to Manipulate*/
/*----------*/
execute “C:\windows\systems\notepad.exe”

}
}

Appendix G : Partial Extraction (Re-encode) Level 2

The algorithm of partial extraction level 2 (re-encode) process for error correction level L.

```
/* Algorithm Module Index P23 – Initialisation */
/*-------------------------------------------------*/

/* Creating the package to be used */
package qrcodecmm;

/* Initialisation java library */
call import java.io.FileInputStream;
call import java.io.FileOutputStream;
call import it.sauronsoftware.base64.Base64;
call import java.io.IOException;

/* Creating main class */
public class QRCodeCMM {

/* Initialisation variables to be used */
initialize static String plainTextFile = "D:/QR
Code/QRCode/Journal3/singleTextLevel1L.txt";
initialize static String gZipTextFile = "D:/QR
Code/QRCode/Journal3/gZipTextFile.gzip";
initialize static String base64TextFile = "D:/QR
Code/QRCode/Journal3/base64TextFile.b64";
initialize static String filePath = "D:/QR Code/QRCode/Journal3/";
initialize static String fileName = "fileNumber";
initialize static String fileType = ".txt";
initialize static String fileTypePNG = "png";
initialize static String fileRGB = filePath + "fileRGB." + fileTypePNG;
initialize static final int size = 551;

/* Creating main programme */
public static void main(String[] args) {

/* Counting the total characters including line feed and carriage return */
initialize object CounterLetters count = new CounterLetters();
execute count.CountLetter(plainTextFile);

/* Algorithm Module Index P24 – Compression utility (GZip) */
/*-------------------------------------------------*/

initialize object GZip gZip = new GZip();
```

```
execute gZip.gZipFile(plainTextFile, gZipTextFile);

/* Algorithm Module Index P25 – Encoder for Base64*/
/*----------------------------------------------------------------*/

initialize object base91cli base91 = new base91cli();
try {
initialize object FileInputStream ifs = new FileInputStream(gZipTextFile);
initialize object FileOutputStream ofs = new FileOutputStream(base64TextFile);
execute base91.encode(ifs, ofs);
} catch (Exception e) {
display error System.err.println(e);
}

/* Algorithm Module Index P26 – Compatibility QR Code ANSI to UTF */
/*----------------------------------------------------------------*/

initialize object ansiToUTF8 utf8 = new ansiToUTF8();
execute utf8.convert(base64TextFile);

/* Algorithm Module Index P27 – Characters distribution to files */
/*----------------------------------------------------------*/

initialize object CreateQRCode1 create1 = new CreateQRCode1(size);
initialize object CreateQRCode9 create9 = new CreateQRCode9(size);
initialize object CreateQRCode17 create17 = new CreateQRCode17(size);

/* Counting characters */
initialize object  CountChar countChar = new CountChar();
execute int countCharacter = countChar.count(base64TextFile);

/* Divide characters with related value and fit each of file */
initialize object  DivideCharacters divide = new DivideCharacters();

/* Create blank files */
execute create1.createBlank40Files(filePath, fileName, fileType);

/* Embedded text characters  to each blank file created*/
execute int totalFiles = divide.divideCharacter(base64TextFile, countCharacter,
filePath, fileName);

/* Checking if the file exceed more than 8 */
      if (totalFiles >= 8) {
         System.exit();
      }
```

```
/* Specify three group that contains 8 files each */
initialize int eight = 8;
initialize String[] multiColourLayerFail = {"QRRed", "QRGreen", "QRBlue"};
initialize int colourCombineRGB[][][] = new int[size][size][3];
initialize MultiLayerQRCode multiLayerQRCode = new MultiLayerQRCode();

/* Algorithm Module Index P28– Create black and white QR Code and combine
it in a group.  */
/*-----------------------------------------------------------------------------------------------
----------*/

/* Start with first  group (red) */
if (totalFiles >= 0) {
        initialize int total8 = 8;

        /* Create an updated single black and white QR Code (Module Index
P12)
        execute create1.generateQRCodeVersion40(filePath, fileName, fileType,
fileTypePNG, total8);

        /* get information from QR Code group 1 (Module Index P13)
        try {
          execute resultFinal = create1.readImage(filePath, fileName, fileTypePNG,
total8);
        } catch (IOException ex) {
        }
        execute int[][] plotResultBlackWhite =
create1.generateMultiplexQRCode(resultFinal, total8);

        /* Combine pixels among 8 black and white QR Codes */
        for (int x = 0; x < size; x++) {
          for (int y = 0; y < size; y++) {
            execute colourCombineRGB[x][y][0] = plotResultBlackWhite[x][y];
          }
        }

/* Algorithm Module Index P29 – Create monocoloured QR Code.  */
/*-----------------------------------------------------------------------------------------------
----------*/
       execute create1.generateQRCodeVersion40MonoColour(filePath,
fileTypePNG, plotResultBlackWhite, multiColourLayerFail[0]);
}
    }

/* Algorithm Module Index P30– Create coloured QR Code.  */
/*-----------------------------------------------------------------------------------------------*/
```

```
        initialize CombineRGB combineRGBColour = new CombineRGB();
        execute combineRGBColour.combineRGB(colourCombineRGB, fileRGB);
    }
}
```

Appendix H : Processing Time Module

The module of processing time for the current experiments of encoding, decoding and partial extraction.

```
   private static String toString(long nanoSecs) {
      int minutes = (int) (nanoSecs / 60000000000.0);
      int seconds = (int) (nanoSecs / 1000000000.0) - (minutes * 60);
      int millisecs = (int) (((nanoSecs / 1000000000.0) - (seconds + minutes * 60)) *
1000);

      if (minutes == 0 && seconds == 0) {
         return millisecs + "ms";
      } else if (minutes == 0 && millisecs == 0) {
         return seconds + "s";
      } else if (seconds == 0 && millisecs == 0) {
         return minutes + "min";
      } else if (minutes == 0) {
         return seconds + "s " + millisecs + "ms";
      } else if (seconds == 0) {
         return minutes + "min " + millisecs + "ms";
      } else if (millisecs == 0) {
         return minutes + "min " + seconds + "s";
      }
      return minutes + "min " + seconds + "s " + millisecs + "ms";
   }
```